

03 SEP 2004

Helsinki 29.4.2003

REC'D 15 MAY 2003

WIPO PCT

ETUOIKEUSTODISTUS
PRIORITY DOCUMENT



Hakija
Applicant

Nokia Oyj
Helsinki

Patenttihakemus nro
Patent application no

20020414

Tekemispäivä
Filing date

04.03.2002

Kansainvälinen luokka
International class

G06N

Keksinnön nimitys
Title of invention

"Mechanism for unsupervised clustering"
(Mekanismi valvomatonta klusterointia varten)

Täten todistetaan, että oheiset asiakirjat ovat tarkkoja jäljennöksiä Patentti- ja rekisterihallitukselle alkuaan annetuista selityksestä, patenttivaatimuksista, tiivistelmästä ja piirustuksista.

This is to certify that the annexed documents are true copies of the description, claims, abstract and drawings originally filed with the Finnish Patent Office.


Pirjo Kaila
Tutkimussihteeri

**PRIORITY
DOCUMENT**
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH RULE 17.1(a) OR (b)

Maksu 50 €
Fee 50 EUR

Maksu perustuu kauppa- ja teollisuusministeriön antamaan asetukseen 1027/2001 Patentti- ja rekisterihallituksen maksullisista suoritteista muutoksineen.

The fee is based on the Decree with amendments of the Ministry of Trade and Industry No. 1027/2001 concerning the chargeable services of the National Board of Patents and Registration of Finland.

Osoite: Arkadiankatu 6 A Puhelin: 09 6939 500 Telefax: 09 6939 5328
P.O.Box 1160 Telephone: + 358 9 6939 500 Telefax: + 358 9 6939 5328
FIN-00101 Helsinki, FINLAND

BEST AVAILABLE COPY

MECHANISM FOR UNSUPERVISED CLUSTERING

BACKGROUND OF THE INVENTION

The invention relates to clustering techniques that are generally used to classify input data into groups or clusters without prior knowledge of those clusters. More particularly, the invention relates to methods and apparatus for automatically determining the cluster centres. An example of such clustering techniques is a Self-Organizing Map, originally invented by Teuvo Kohonen. The SOM concept is well documented, and a representative example of a SOM application is disclosed in US patent 6 260 036.

The current framework under investigation for describing and analyzing context has a critical component based on clustering of data. This clustering is expected to appear at every stage of context computation, from the processing of raw input signals to the determination of higher order context. Clustering has been well studied over many years and many different approaches to the problem exist. One of the main problems is knowing how many clusters exist in the data. Techniques exist to estimate the number of clusters in a data set, however the methods either require some form of a priori information or assumptions on the data, or they estimate the number of clusters on the basis of an analysis of the data, which may require storing the data, as well as being computationally demanding. None of these approaches seem entirely suitable for on-line, unsupervised cluster analysis in a system with limited resources as would be the case for a context aware mobile terminal. With this in mind, and in what follows an algorithm for on-line cluster analysis which is computationally simple and needs no assumption about the number of clusters in the data being analyzed, is described.

Clustering is an important part of any data analysis or information processing problem. The idea is to divide a data set into meaningful subsets so that points in any subset are closely related to each other and not related to points in other subsets. The definition of related may be as simple as the distance between points. Many different approaches and techniques can be applied to achieve this goal. Each approach has its own assumptions and advantages and disadvantages. One of the best-known methods from the partition-based clustering class is the K-means algorithm, which adaptively tries to position K 'centres' which minimize the distance between the input data vectors and the centres. One of its disadvantages is that the number of the K

centres must be specified before the clustering is attempted. In the case of an unknown data set this may not always be possible. The algorithm can be run several times with different values of K and the optimum K is chosen on the basis of some criteria. For an on-line system where the data is not stored, this approach is slow and impractical.

Thus a problem associated with the known clustering techniques is that while it is relatively easy for humans to determine the cluster centres, such a determination is difficult for computers.

BRIEF DESCRIPTION OF THE INVENTION

An object of the invention is to provide a method and an apparatus for implementing the method so as to alleviate the above disadvantages. In other words, the object of the invention is to provide a method for automatically determining the cluster centres, such that the method is easily implemented in a computer system.

The object of the invention is achieved by a method and an arrangement which are characterized by what is stated in the independent claims. The preferred embodiments of the invention are disclosed in the dependent claims.

A computer-implemented method according to the invention can be implemented by the following steps:

initializing a first data structure that comprises a lattice structure of weight vectors that create an approximate representation of a plurality of input data points;

performing a first iterative process for iteratively updating the weight vectors such that they move toward cluster centres;

performing a second iterative process for iteratively updating a second data structure utilizing results of the iterative updating of the first data structure; and

determining the weight vectors that correspond to cluster centres of the input data points on the basis of the second data structure.

A preferred embodiment of the invention is based on the following idea. Self-organizing maps generally use a lattice structure of nodes and associated with each node is a weight vector. Each data point in the input data is iteratively compared with each weight vector of the lattice, and the node whose weight vector best approximates the data point is chosen as the winner

for that data point and iteration. Then the weight vectors associated with each node of the lattice are adjusted. The adjustment affected to each node's weight vector is dependent on the winning node through a neighbourhood function. Following the adjustment of the weight vectors a next iteration step is taken.

5 As used in this context, the term 'neighbourhood function' is a function of distance on the lattice between the winning node and the node being updated such that the value of the function generally decreases as the distance increases. With normalized SOMs, the value of the function is unity for a distance of zero. A common form for the neighbourhood function is
10 Gaussian, but preferred embodiments of the invention make use of neighbourhood functions that are not strictly Gaussian.

 In addition to the primary iteration process for updating the SOM or other clustering mechanism, a second iterative process is run, and the second iterative process gives a numerical value for the lattice nodes such that the
15 numerical value increases if the node's weight vector is positioned at a cluster centre. Then the cluster centres are determined, not on the basis of the weight vectors, but on the basis of the numerical values produced by the second iterative process.

 Thus the problem of locating cluster centres reduces to a relatively
20 straightforward problem of locating local maxima in the numerical values produced by the second iterative process.

 An advantage of the invention is that it is much easier for machines to locate local maxima in the numerical values than to locate cluster centres in the clustering mechanism wherein the cluster centres are the location in which
25 the density of the weight vectors is highest.

 In a preferred embodiment of the invention, the second data structure comprises a coefficient for each of the weight vectors in the lattice structure. Each iteration in the first iterative process comprises selecting a winner weight vector for each of the data points on the basis of a distance
30 measure between the input data point and the weight vector. Each iteration in the second iterative process comprises calculating a next value of each coefficient on the basis of the current value of the coefficient; and a combination of: 1) the current coefficient of the winner weight vector, 2) a second neighbourhood function that approaches zero as the distance on the
35 lattice structure between the weight vector and the winner weight vector increases, and 3) an adjustment factor for adjusting convergence speed

between iterations.

The combination referred to above can be a simple multiplication.

If the second neighbourhood function is selected appropriately, such that the second data structure has distinct borders, the step of determining the weight vectors can be accomplished simply by selecting local maxima in the second data structure.

A preferred version of the second neighbourhood function is not monotonous, giving negative values at some distances. Also, the second neighbourhood function is preferably made more pronounced over time, as the number of prior iterations increases.

Preferably, the first data structure is or comprises a self-organizing map and the input data points represent real-world quantities.

BRIEF DESCRIPTION OF THE DRAWINGS

In the following the invention will be described in greater detail by means of preferred embodiments with reference to the attached drawings, in which

Figure 1 illustrates a self-organizing map (SOM) with six clusters of input data points;

Figure 2 shows a typical form of the neighbourhood function used in a SOM algorithm;

Figure 3 shows a 15 by 15 lattice structure resulting from uniformly-distributed input data;

Figure 4 shows a probabilistic map for visualizing the cluster centres in a SOM with six cluster centres;

Figure 5 shows a preferred form of the second neighbourhood function used in the second iterative process according to a preferred embodiment of the invention;

Figure 6 shows a computer pseudocode listing for generating the function shown in Figure 5.

Figure 7 shows a coefficient map that visualizes the data structure used for locating the cluster centres in the SOM; and

Figure 8 is a flow chart illustrating a method according to the invention wherein the method comprises two iterative processes run in tandem;

Figures 9 and 10 show another pair of SOM map and coefficient

map, respectively, with five clusters;

Figures 11 and 12 show a SOM and coefficient map, respectively, for an exceptional distribution of input data;

Figure 13 shows an example of a neighbourhood function used in an automatic cluster-labelling algorithm;

Figure 14 shows the result of the automatic cluster-labelling algorithm; and

Figure 15 shows how the automatic cluster-labelling algorithm can be integrated with the cluster-determination algorithm according to the invention.

DETAILED DESCRIPTION OF THE INVENTION

A practical example of the invention is disclosed in the context of self-organizing maps. A SOM is a learning algorithm or mechanism from the area of Artificial Neural Networks (ANNs) that find wide application in the area of vector quantization, unsupervised clustering and supervised classification. Reasons for its widespread use include its robustness, even for data sets of very different and even unknown origin, as well as the simplicity of its implementation. The SOM uses a set of weight vectors to form a quantized, topology-preserving mapping, of the input space. The distribution of the weights reflects the probability distribution of the input. The SOM representation is used in clustering applications generally by clustering the weight vectors after training, using for example the K-means algorithm. However the problem of the original K-means algorithm still remains, that is, determining the value of K the number of centres. In the following a method, based on the SOM algorithm, is described which can be used to automatically determine cluster centres in an unsupervised manner. That is the number of clusters does not have to be predefined and groups of adjacent SOM weight vectors represent the cluster centres. Unlike the K-means algorithm where each cluster is represented by one centre, in the inventive algorithm the cluster is represented by a set of centres which correspond to weight vectors in the SOM. The algorithm requires few additional computational resources and makes direct use of the information generated during the learning of the SOM. It is already clear how the algorithm can be considered a hybrid of the K-means algorithm and a probabilistic mixture model based method. Each cluster is represented by a set of centres, which correspond to a set of weights in the

SOM. In turn the SOM weights form an approximation of the probability distribution of the input.

From the ANN point of view it may be interesting as the algorithm uses lateral inhibition between the weight vectors to generate the clusters and a form of Hebbian learning. It is clear that the performance of the clustering is heavily dependent on the topology preserving and convergence ability of the SOM.

The SOM algorithm

Figure 1 shows a self-organizing map 10. More particularly, Figure 1 shows an online version of SOM. There also exists a "Batch SOM" but this requires storing all the input points and going through all of them several times which is why the online-version is preferred here. Reference numerals 11 generally denote input data points that in most SOM applications represent real-world events or quantities. The SOM algorithm creates a SOM or lattice structure 12 by means of an iterative process that can be summarized as follows. Consider a time sequence of inputs $\omega(t)$, $t = 1, \dots$, with $\omega \in R^m$ and a probability distribution p_ω . The SOM itself consists of a total of N weight vectors $X \in R^m$ distributed on an n -dimensional lattice. Thus there are two associated dimensions, the dimension m of the input data and the weight vectors, and the dimension n of the lattice. The reason for having a lattice is to be able to define neighbourhood relations between adjacent weights. For example, if each weight k has an associated position vector $i_k \in I^n$ on the lattice, then the distance $d_I(i_j, i_k)$ between weights k and j on the lattice can be defined. The initial values of the weight vectors can be randomly chosen, as the convergence of the algorithm is independent of the initial conditions. At each iteration, the distance $d(\omega(t), X_k)$ between the input $\omega(t)$ at time t and each of the weight vectors X_k is calculated. The winner weight $v(t)$ at time t is then defined as:

$$v(t) = \arg \min_{1 \leq k \leq N} d(\omega(t), X_k) \quad [1]$$

For real-valued data, a possible distance measure $d(\cdot, \cdot)$ could be the Euclidean distance. When the winner has been found each weight vector is then updated as:

$$X_k(t+1) = X_k(t) + \alpha(t)h(k, v(t))(\omega(t) - X_k(t)) \quad [2]$$

where $\alpha(t) \in [0,1]$ is a decreasing function of time which determines the learning rate of the SOM and h is a neighbourhood function which is a function of the distance on the lattice between the winner weight and the updated weight, that is: $h(k, v(t)) = h(d_L(i_{v(t)}, i_k))$.

Figure 2 shows a typical form of the neighbourhood function h used in the SOM algorithm. The function h is positive, having a maximum value of 1 for $d_L = 0$ and decreasing monotonically towards 0 as d_L increases. In practical situations h is often Gaussian, i.e., of the form:

$$h(d_L) = e^{-ad_L^2} \quad [3]$$

... for some appropriately chosen scaling factor a which may be a function of time. The SOM map is built iteratively. In other words, the steps in equations [1] and [2] are repeated for each t . For uniformly-distributed input data and a two-dimensional lattice with 15×15 weights, the SOM algorithm gives a result as in Figure 3.

Figure 3 shows a plot of the weight values on the support $[0,1] \times [0,1]$ of the input, where adjacent weights on the lattice are connected by a line segment in the plot. None of the line segments intersect, which indicates that the weights are organized. The weight values are spread uniformly over the input space and approximate the uniform distribution of the input data. To arrive at such a result the learning is divided into two phases. In the first phase, there is a large α and a , and the algorithm is in the self-organizing phase where the weights become organized. The second phase is the convergence phase where the neighbourhood width is reduced to zero as a increases and α approaches zero. The weight vectors converge to the approximation of the probability distribution of the input.

Unsupervised Clustering Based on the SOM

After the description of the basic SOM algorithm, some techniques of determining cluster centres will be disclosed. Consider the SOM shown in Figure 1. Figure 1 shows data points generated from six different normal distributions with different means and variances and plotted on this are the weight vectors from the SOM when data from this distribution was used as an input. It is seen that the weight vectors have organized and are more concentrated at the centres of the clusters. Hence the approximation of the probability distribution of the input. Reference numerals 13_1 to 13_6 denote six circles located at the cluster centres. The circles are not normally part of the

SOM. While it is easy for humans to determine where the cluster centres are, this task is surprisingly difficult for computers.

First, a probabilistic algorithm for determining the cluster centres will be briefly disclosed. Figure 4 shows a plot of the calculated probability of each weight vector being chosen as winner during the simulation. The probability for each weight vector is determined by keeping a record of the number of times the weight vector was chosen as winner and then dividing the number by the total number of iterations in the simulation. The i, j axis indicate the position of each weight on the SOM lattice and the $p(i, j)$ axis shows the probability of each of the weights. From this probability distribution the cluster structure is visible, where local maximum in the surface correspond to weight vectors positioned at input data cluster centres and the local minimum form a boundary between the local maximum and hence the clusters. However, the different clusters are not clearly distinct and the surface is not smooth. This roughness of the surface and variations in the peak values of the local maximum can lead to problems when trying to automatically determine the cluster centres using an algorithm which would associate a local maximum of the probability with a cluster centre. Defining a global threshold above which a weight vector would be considered a local maximum and a second global threshold below which a weight vector would be considered a local minimum leads to problems. For example, if in one part of the lattice the probability at a local maximum is below the global threshold to be considered a local maximum hence a cluster centre could lead to erroneous choice of cluster centres. In practise, as in the example shown, this is likely to occur. The alternative would be to define local thresholds for determining local maximum and minimum. However this would be a complicated process requiring an involved computation with no guarantee of a correct result.

A clustering algorithm according to a preferred embodiment of the invention is based on this observation. In effect the algorithm provides a means to smooth the probability surface just described. The use of a neighbourhood function means that the smoothing operation is done locally, emphasising local maximum as well as local minimum. The positive elements of the neighbourhood function emphasise the local maximum and the negative components emphasise the local minimum. The result is that the local maximums all reach consistently high values and local minimum reach consistently low values. This allows for the use of a global threshold to identify

the maximum and minimum and hence facilitating the use of a computer in the process. Hence instead of using directly the probability of a weight being the winner, a measure somehow related is used in the proposed algorithm which is described as follows.

5 For each weight i define a scalar coefficient C_i . This coefficient is bounded to the interval $[0, 1]$ and its initial value before training can be quite small. The SOM algorithm is carried out as described earlier. At each iteration the winner weight $v(t)$ is determined as in equation [1] and each SOM weight i is updated according to equation [2]. At the same time each coefficient C_i is
10 updated as:

$$C_i(t+1) = C_i(t) + C_{v(t)}(t)h_m(d_L)\delta \quad [4]$$

where h_m is the second neighbourhood function. d_L and $v(t)$ are the same terms that were used in the SOM algorithm, that is, d_L is the distance on the lattice between node i and the winner node $v(t)$. δ is a small step value for
15 adjusting convergence speed. δ is somewhat analogous to the α in the SOM algorithm.

Following the update then $C_i(t+1)$ is forced within the interval $[0, 1]$. For example, if $C_i(t+1) > 1$, it can be set to 1, and if $C_i(t+1) \leq 0$, it can be set to 0.01. The Hebbian-type learning is clear as the update of $C_i(t)$ depends on
20 the value of $C_{v(t)}$.

Figure 5 shows a preferred form 51 of the second neighbourhood function h_m . Actually, Figure 5 shows a time-dependent version of the second neighbourhood function h_m such that the function is more pronounced as the number of prior iterations increases. In other words, with increasing time, the
25 function h_m achieves negative values sooner (as the distance d_L increases), and the negative values are much more negative than during the earlier iterations.

As can be seen, the preferred form 51 of the second neighbourhood function h_m somewhat resembles the first neighbourhood function h used in the
30 SOM algorithm. Like the first neighbourhood function h , the second neighbourhood function h_m starts at 1 when the distance is zero. Also, h and h_m both approach zero as the distance d_L increases.

But the second neighbourhood function h_m is preferably negative for some distances. For instance, in a 10 by 10 lattice, h_m may be negative for
35 distances over 3. The negative value of the second neighbourhood function h_m

can be seen as a form of lateral inhibition between the weights. Lateral inhibition is a mathematical model that tries to approximate real biological phenomena. Similar to the h function used in the SOM, weights adjacent to the centre of activity have their coefficients and hence their activity increased, while the activity of weights further away from the centre of activity are inhibited.

However this lateral inhibition is rarely if ever used in practical applications. In the SOM the interaction between the weight vectors is defined by the neighbourhood function h defined by equation [3] which is strictly positive. If h was allowed to be negative at any point, divergence of the weight vectors could result, instead of convergence. In the clustering method proposed here this lateral inhibition is used to determine the cluster centres.

Intuitively it can be seen that if a weight i is quite often the winner then C_i will increase along with its neighbours. At the same time for i the winner, and for j far from i on the lattice C_j will be decreased. Similarly, if i is not often the winner its C_i will not increase very much and will be decreased by other winners at a distance on the lattice. Given the example in Figure 1 it is clear that weights in the inter-cluster regions will have a lower probability of being winners, whereas weights close to the cluster centres will have a higher probability of being winners. Hence it would be expected that the coefficients C_i would be larger for weights positioned in or near the cluster centres and small for positions between clusters. This would then provide boundaries between the cluster centres. Of course this depends on the fact that the weight vectors X_i reach an organized configuration.

Figure 6 shows an example of a computer pseudocode listing for generating the function shown in Figure 5.

Figure 7 shows a plot of the coefficient values $C(i, j)$ for the weights i, j in the SOM of Figure 1 with $\delta = 0.01$ after 20000 iterations. The second neighbourhood function h_m varied with time and initially did not have a high level of lateral inhibition. Towards the end of the simulation, the level of lateral inhibition was increased. The surface shown in Figure 7 has six distinct and separated elevated regions. As a result of forcing the C_i between 0 and 1, each elevated region corresponds to a set of adjacent weights on the lattice whose coefficients C_i have saturated at or near 1 and whose weights X_i are found at the cluster centre. These regions are surrounded by regions where the coefficients C_i have been driven to small values. Hence it is possible to

determine which weights represent the same cluster. To determine which cluster an input vector belongs to, simply find the closest centre and assign the input to the cluster to which the centre belongs. Thus unlike the K-means algorithm where one weight represents the cluster, in the algorithm proposed here a group of weights represents the cluster. The means of classification is the same. Because the coefficients C_i are saturated near 0 or 1, it is a simple task to determine the cluster centres using a global threshold which could be set for example at a value of 0.5.

It should be noted that the plot 70 is for visualization purposes only and is not required by computers. Instead, reference number 71 points to an array of current coefficients $C_i(t)$ and reference number 72 points to an array of next coefficients $C_i(t+1)$. It is the array 72 of updated coefficients that a computer uses to determine the cluster centres and their locations. The arrow 806 denotes updating of the coefficients that takes place in step 806 of Figure 8 that will be described next.

Figure 8 is a flow chart illustrating a method according to a preferred embodiment of the invention wherein the method comprises two iterative processes 81 and 82 run in tandem. The odd-numbered steps on the left-hand side of Figure 8 relate to the known SOM algorithm. The even-numbered steps on the right-hand side of Figure 8 relate to the inventive algorithm for maintaining and updating the second data structure that is used to determine the cluster centres automatically. In step 801 the SOM algorithm is initialized. The initialization comprises selecting initial values for the α , a , h and randomly initializing the weight vectors X_i . Since Figure 8 shows an online algorithm, the values of the inputs $\omega(t)$ are not known at this stage. Step 802 is a corresponding initialization step for the second data structure. Steps 803, 805, 807 and 809 form the conventional SOM iteration. In step 803, input $\omega(t)$ is presented to the SOM at iteration t . In step 805, the winner weight $v(t)$ is selected according to equation [1]. In step 807, the weight vectors are updated according to equation [2]. In an optional step 807', the variables α that determines learning speed and/or the a of the first neighbourhood function h are updated. In step 809, the iterative loop is repeated until some predetermined stopping criteria have been met. For instance, the loop may be set to run a predetermined number of times, or the loop may be interrupted when each succeeding iteration fails to produce a change that exceeds a given threshold.

Steps 806 and 808 relate to the second iterative process 82 for maintaining and updating the second data structure that is used to determine the cluster centres. In step 806, the coefficients C_i are updated on the basis of the winner weights $v(t)$ according to equation [4]. In an optional step 808, parameters for the second neighbourhood function h_m are updated (see Figure 5). According to a preferred feature of the invention, steps 806 and 808 of the second iterative process 82 are interleaved with the steps of the first iterative process 81. In this way, step 806 utilizes intermediate calculation results of step 805. Similarly, step 808, in which the second neighbourhood function h_m is updated, may utilize data from step 807 that updates the variable α and the first neighbourhood function.

Figures 9 and 10 show another pair of SOM map and coefficient map, respectively. In this example the number of clusters in the input data distribution was reduced by one and the same SOM algorithm was used. Figure 9 shows the result of both the input and the final configuration of the weight vectors. Figure 10 shows the resulting C_i values. The five cluster centres are once again quite clear. It is remarkable that the same algorithm, without any adjustments, was capable of finding the new number of clusters and the locations of those clusters.

The way the invention works is as follows. In the beginning there is a predefined lattice, which in this case is a two-dimensional. Each point of the lattice is given a label, e.g. (2,3), (0,15). This lattice remains fixed and the labelling of the lattice points does not change. In the above examples, the lattice structure is a 15×15 lattice. It is from the lattice that the distances dL used in all neighbourhood functions is determined, e.g. the distance between lattice points (1,3) and (7,8) could be 6, depending on which distance measure we use.

Each lattice point is associated with a weight vector. The dimension of the weight vector is always the same dimension as the input data vector. In the examples here the input data has two dimensions. It is the weight vectors that change depending on the input data point and the distance, on the lattice, between two lattice points, the first point being the lattice point associated with the winner and the second point, the lattice point associated with the weight vector to be updated. This distance is not used to update the weight vector directly, but is used to determine the value of the first neighbourhood function in the update of the weight vector.

Figures 1 and 7 show a two-dimensional plot of the input data points and the weight vectors from the SOM. The plot is in the input space and by chance all the input points were bounded by $[-8, 8]$. The data points are just the points shown. The weight vectors are plotted at the crossing point of the lines. Another way of looking at it is, a line is drawn between weight vectors whose associated lattice nodes are the closest nodes on the lattice. The fact that the plot appears as a lattice means that the weights are organized, that is, the weight vector associated with adjacent lattice nodes appear in the input data space as adjacent to each other.

The relationship between the coefficients and the SOM lattice is the same as the relationship between the weight vectors and the SOM lattice, except the dimension of the coefficients is always 1. It is somewhat similar, though not the same as a probability measure, where the probability would be that the weight vector associated to the lattice node to which the coefficient is associated will be chosen as the winner for any given data input. Another interpretation is that the coefficients somehow represent an exaggerated version of the probability distribution of the input data.

In conclusion, we might say that the lattice is a fixed structure. One weight vector is associated with each lattice node. The weight vector is in the input data space. Similarly, one coefficient is associated with each lattice node. It is a scalar value and represents an indication of probability, though not the real probability that the weight vector associated with the same lattice node will be chosen as winner for a given input data point.

The dimensionality of the input data and the lattice are not necessarily the same. The input data may have any number of dimensions, such as 5, 10 or 15. In that case the dimension of the weight vectors would also be 5, 10, or 15. But the lattice could still be two-dimensional. We could also choose a different lattice to begin with (i.e., change the SOM structure) and make it four-dimensional, for example. In this case, if we still choose to have 15 lattice nodes along each axis of the lattice then we would have $15 \times 15 \times 15 \times 15$ lattice nodes and associated with each lattice node a 5, 10 or 15-dimensional weight vector. The examples above use a two-dimensional lattice and a two-dimensional input space merely because it is easier to draw and visualize. In practical implementations one could expect that the input data has more dimensions but the lattice structure could be two-dimensional.. The

number of lattice nodes along each dimension of the lattice is variable depending on the amount of computational resources available.

Figure 11 shows a mixed clustering example consisting of two very different clusters, namely a first cluster 112 described by a normal distribution and a second cluster 114 described by a uniform distribution along a parabola. The same SOM was used as in the previous two examples. Figure 12 shows the map 120 of coefficients C_{ij} for this example. There are two distinct areas. There is a connected region 122 of high values around three edges of the lattice which correspond to the parabolic distribution cluster 112 in Figure 11, and the disc shape 124 of connected values which corresponds to the normal distribution 114. This example is quite interesting because of the complexity of the parabolic distribution. In the mixture model clustering algorithm, without any prior information it would be difficult to generalize this distribution to a normal distribution. Similarly, a K-means approximation would have difficulty resolving these two clusters as at some points the distance between two points in different clusters are smaller than between two points in the same cluster. In experiments on this example the best K-means came up with four clusters. This example indicates that the clustering algorithm according to the invention can be very general.

Automatic labelling of cluster centres

A further preferred embodiment of the invention relates to automatic and unsupervised labelling of the clusters. The same notation is used here as above and only notation pertinent to this embodiment will be explained. Consider a set of labels $B = \{1, 2, \dots, K\}$ which will be used to label the clusters. In practise K should be at least greater than or equal to the expected number of clusters. In the case of no prior knowledge, it may be suitable to let $K = N$ the total number of weights in the SOM, as this imposes a limit on the maximum number of clusters which can be identified.

For each weight i in the SOM define a vector of coefficients Θ_i as:

$$\Theta_i = (\theta_{i,1}, \theta_{i,2}, \dots, \theta_{i,K}) \quad [5]$$

Each coefficient $\theta_{i,l} \in [0, 1]$ represents a weighting between the SOM node i and the label l . The weight i belongs to cluster l if:

$$l = \arg \max_{1 \leq k \leq K} \theta_{i,k} \quad [6]$$

The updating algorithm used on these coefficients to achieve automatic labelling proceeds as follows. At time t SOM weight $v(t)$ is chosen as the winner. The weight and its neighbours are updated as in the normal SOM algorithm. Also the coefficient C_i is updated as well as the coefficients C_j of the neighbours of C_i . The updating of the coefficients C_i and the interpretation of the results form the basis of the main invention, namely the automatic and unsupervised clustering.

In the automatic and unsupervised labelling of the clusters, at the same time t the Θ_i are updated as follows. Define $lv(t)$ as the label of the cluster to which the winner weight $v(t)$ is assigned, thus from equation (6),

$$lv(t) = \arg \max_{1 \leq k \leq K} \theta_{v(t),k}(t) \quad [7]$$

For all the weights j , $j = 1, \dots, N$ then the components $\theta_{j,lv(t)}$ are updated as follows:

$$\theta_{j,lv(t)}(t+1) = \theta_{j,lv(t)}(t) + C_{v(t)}(t+1) h_B(d_L) \delta \quad [8]$$

where once again h_B is a neighbourhood function and preferably has the form shown in Figure 13. The idea is that the neighbours of the winning weight will have their $\theta_{j,lv(t)}$ increased to ensure that they will be classified to the same cluster as the winner $v(t)$, and weights further away from the winner weight will have their $\theta_{j,lv(t)}$ decreased to ensure that they will be classified to a different cluster than the winner.

For the weights j where the neighbourhood function $h_B(d_L) > 0$ it is also advantageous to decrease the other coefficients $\theta_{j,k}(t)$, $k = 1, \dots, K$, $k \neq lv(t)$ in Θ_j as follows:

$$\theta_{j,k}(t+1) = \theta_{j,k}(t) - C_{v(t)}(t+1) h_B(d_L) \delta \quad [9]$$

This reinforces the labelling of the winner and its neighbours to the cluster label $lv(t)$.

Note that equation [4] uses C_v at iteration t whereas equations [8] and [9] use C_v at iteration $t+1$. Actually, the C_v coefficient changes so little between iterations that either value can be used, depending on which value is more conveniently available.

Figure 14 shows an example of the result of the combination of the SOM, automatic clustering and automatic labelling of the clusters for the case of an input distribution consisting of five normal distributions. In this case the set of labels was given by $L = \{1, 2, 3, 4, 5, 6, 7, 8\}$. The values of the θ_{ij} were

randomly initialized. Figure 14 shows a SOM with five distinct regions, one region around each cluster. Each node in the five areas 1 to 5 is assigned a label such that nodes in area 1 have a label of 2, nodes in area 2 have a label of 4, etc. The inter-region areas have a label of 0. These weights had a maximum of Θ_i less than a threshold value of 0.2 and where not assigned as centres to any cluster. It is clear that the cluster centres have been properly labelled with the labels {2, 4, 5, 6, 8}.

Figure 15 shows how the automatic cluster-labelling algorithm can be integrated with the cluster-determination algorithm according to the invention. Figure 15 is a modification of Figure 8, with step 806 followed by step 806' that relates to the automatic cluster-labelling algorithm. In step 806, the cluster labels $lv(t)$ are determined according to equation [7]. Then the $\theta_{j,lv(t)}$ components are updated according to equation [8] and the $\theta_{j,k}$ components are updated according to equation [9]. By placing the step 806' inside the second iterative process 82, maximal synergy benefits are obtained, or in other words, the computational overhead is kept to a minimum because step 806' makes use of the winner selection and coefficient determination already performed for the SOM construction and the automatic cluster determination.

Summary

The technique according to the invention allows automatic determination of cluster centres with a minimal amount of information on the data. No explicit, initial estimate of the number of clusters is required. Given the nature of convergence of the SOM, there is no need to know the type of distribution of the clusters either. In this respect the algorithm is very general. However, although explicit initial estimates of the number of clusters are not required, care should be taken to ensure that the SOM lattice contains a number of nodes larger than the expected number of clusters, as well as choosing a non-monotonous neighbourhood function that is negative for large distances and provides a level of lateral inhibition to ensure the coefficients for the cluster regions stand out more clearly.

The preferred embodiment of the invention, in which the second iterative process is interleaved with the conventional iterative process, requires little computational overhead. Thus this embodiment invention is especially suitable for on-line application where human supervision is not available. Initial simulations on artificial data show that the inventive technique is simple and

apparently robust and is more easily generalized than most current clustering algorithms. The technique according to the invention can be considered somewhat as a hybrid of the K-means and probabilistic-model-based clustering.

- 5 It is readily apparent to a person skilled in the art that, as the technology advances, the inventive concept can be implemented in various ways. The invention and its embodiments are not limited to the examples described above but may vary within the scope of the claims.

CLAIMS

1. A computer-implemented method for determining cluster centres (13₁ - 13₆) in a first data structure (10), wherein the first data structure comprises a lattice structure (12) of weight vectors that create an approximate representation of a plurality of input data points (11);

the method comprising:

performing a first iterative process (81) for iteratively updating the weight vectors such that they move toward cluster centres (13₁ - 13₆);

performing a second iterative process (82) for iteratively updating a second data structure (70 - 72) utilizing results of the iterative updating of the first data structure; and

determining the weight vectors that correspond to cluster centres of the input data points on the basis of the second data structure (70 - 72).

2. A method according to claim 1, wherein each iteration in the first iterative process (81) comprises:

selecting a winner weight vector (v) for each data point on the basis of the distance between the data point and the weight vectors; and

calculating a next value of each weight vector on the basis of the current value of the weight vector and a first neighbourhood function (21, h) of the distance on the lattice structure between the weight vector and the winner weight vector; and

the second data structure (70 - 72) comprises a first coefficient (C_i) for each of the weight vectors in the lattice structure and each iteration in the second iterative process (82) comprises calculating (806) a next value of each first coefficient (C_i) on the basis of:

the current value of the first coefficient; and a combination of:

a first coefficient of the winner weight vector (v),

a second neighbourhood function (51, h_m) of the distance on the lattice structure between the weight vector and the winner weight vector, and

an adjustment factor (δ) for adjusting convergence speed between iterations.

3. A method according to claim 1 or 2, wherein the step of determining the weight vectors that correspond to cluster centres comprises selecting local maxima in the second data structure (70 - 72).

4. A method according to claim 2 or 3, wherein the combination is or comprises multiplication.

5. A method according to any one of claims 2 to 4, wherein the second neighbourhood function ($51, h_m$) is not monotonous.

5 6. A method according to any one of claims 2 to 5, wherein the first coefficients are limited to the range $[0,1]$ and the second neighbourhood function ($51, h_m$) gives negative or positive values, respectively, for some distances.

10 7. A method according to any one of claims 2 to 6, wherein the second neighbourhood function ($51, h_m$) depends on the number of prior iterations.

8. A method according to any one of the preceding claims, wherein the input data points (11) represent real-world quantities.

15 9. A method according to any one of claims 2 to 8, wherein the first data structure (10) is or comprises a self-organizing map.

10. A method according to claim 9, further comprising:
estimating an upper limit K for the number of clusters in the self-organizing map;
defining a coefficient vector $\Theta_i = (\theta_{i,1}, \theta_{i,2}, \dots, \theta_{i,K})$ for each weight
20 vector i in the self-organizing map, the coefficient vector comprising K second coefficients $\theta_{i,l}$, each of which represents a weighting between the weight vector i and a label l ; and

assigning cluster label l to weight vector i if:
25
$$l = \arg \max_{1 \leq k \leq K} \theta_{i,k}.$$

11. A method according to claim 10, wherein each iteration in the second iterative process (82) comprises calculating (806') a next value of each second coefficient on the basis of the current value of the second coefficient and a combination of:

30 a coefficient of the winner weight vector,
a third neighbourhood function ($131, h_B$) of distance, and

an adjustment factor (δ) for adjusting convergence speed between iterations.

12. A computer-readable program product comprising a computer program code, wherein executing the computer program code in a computer
- 5 causes the computer to carry out the steps of the method according to claim 1.

3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2

(57) ABSTRACT

A computer-implemented method for determining cluster centres in a first data structure, wherein the first data structure comprises a lattice structure of weight vectors that create an approximate representation of a plurality of input data points. The method comprises performing a first iterative process (81) for iteratively updating the weight vectors such that they move toward cluster centres; performing a second iterative process (82) for iteratively updating a second data structure utilizing results of the iterative updating of the first data structure; and determining the weight vectors that correspond to cluster centres of the input data points on the basis of the second data structure.

(Figure 8)

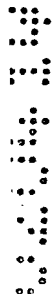


Fig. 1

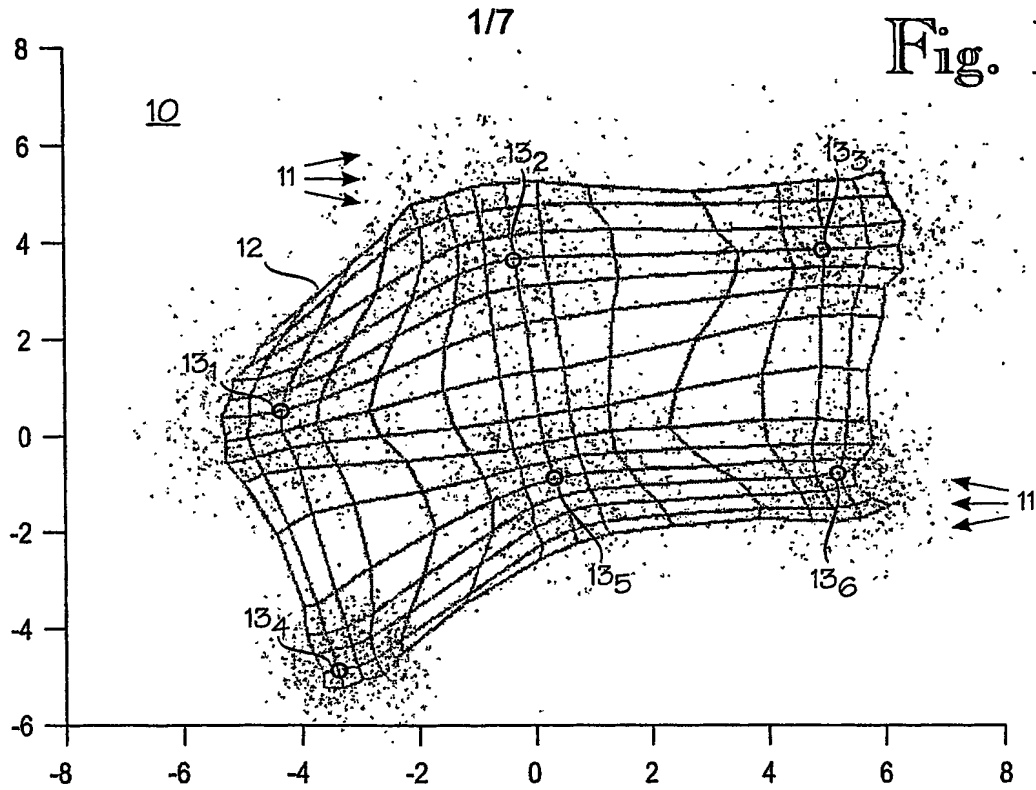


Fig. 2

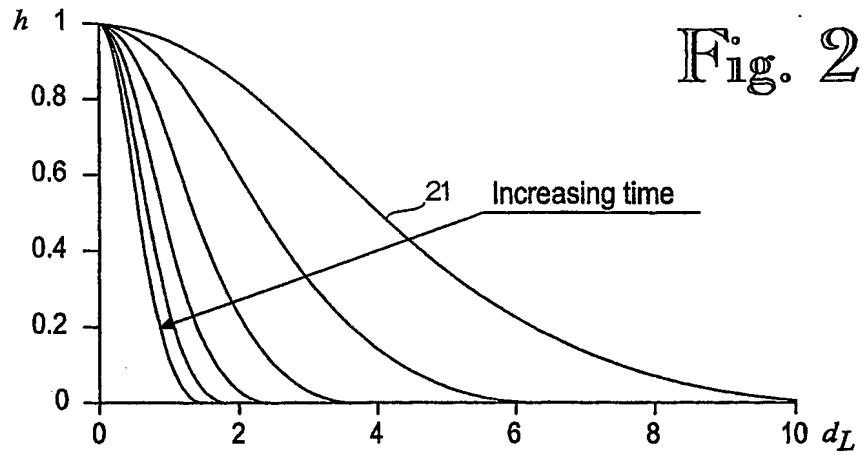


Fig. 3

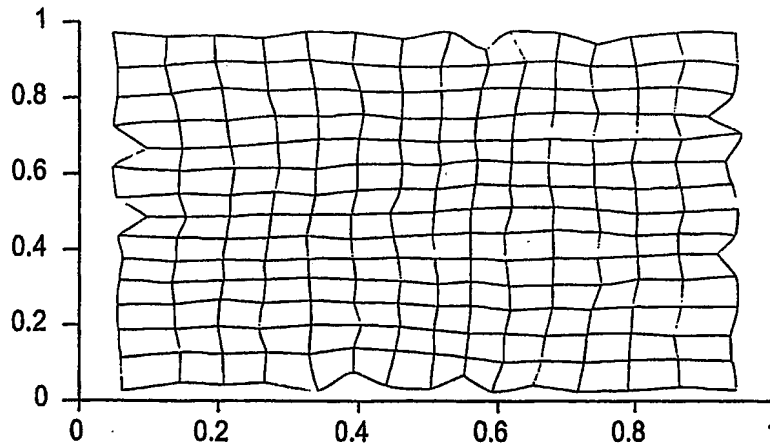


Fig. 6

```

x=linspace(0,10,100);

figure

for i=1:1.5:10
    p=0.3+(0.4*(1-(i/10)));
    y=exp(-0.03*x.*x*i).*(p-(0.01*i*x.*x))/p;
    plot(x,y)
    hold on
end

```

Fig. 7

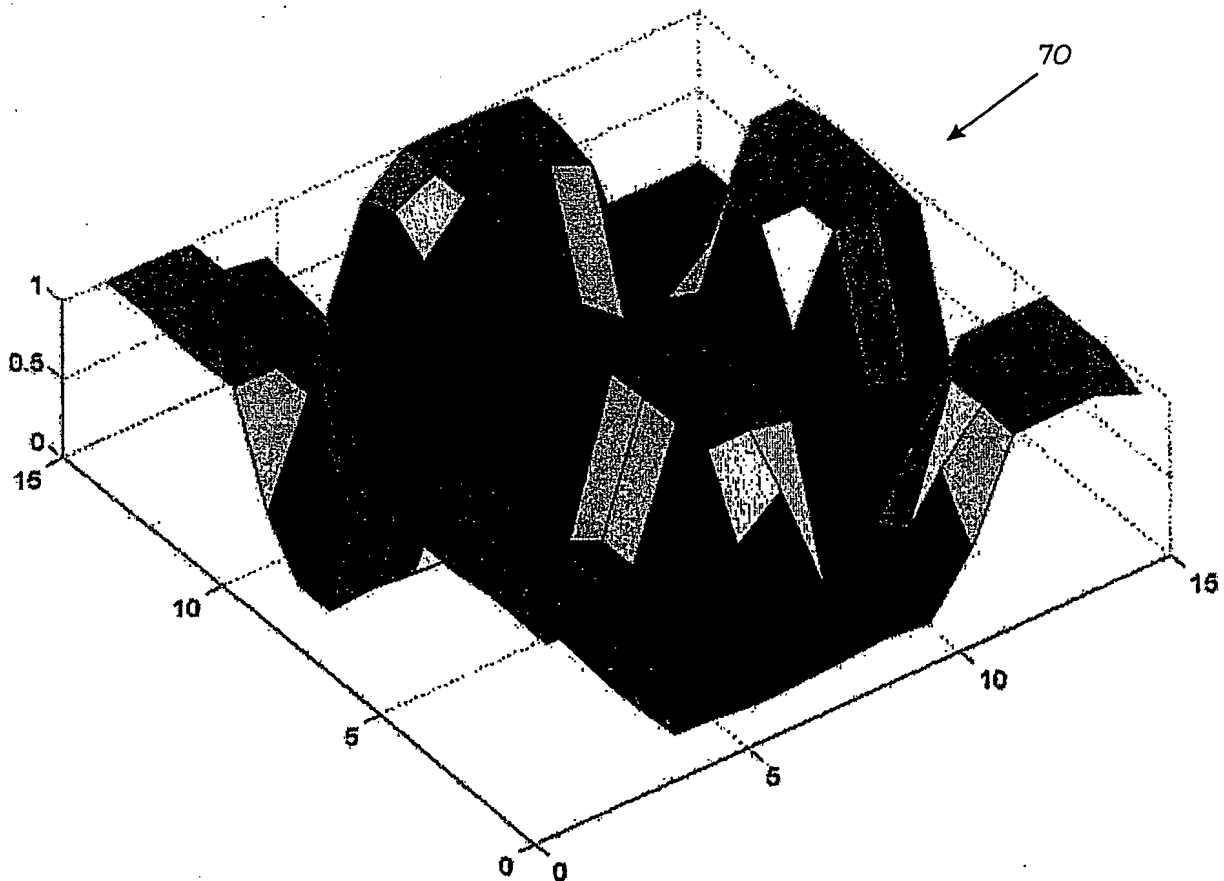
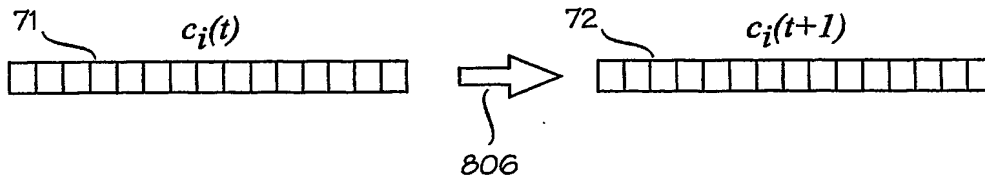


Fig. 8

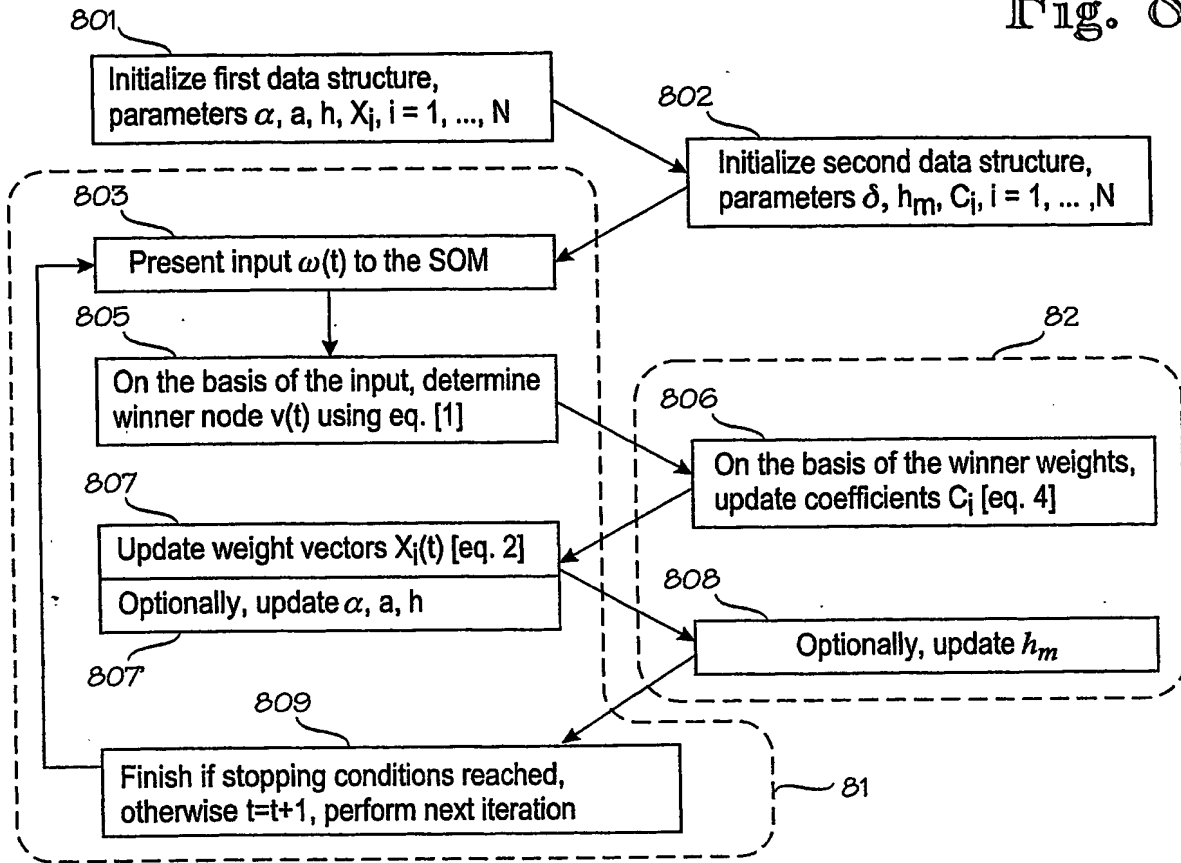
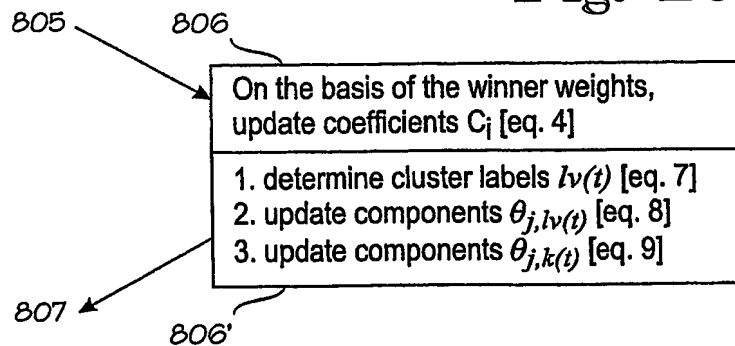


Fig. 15



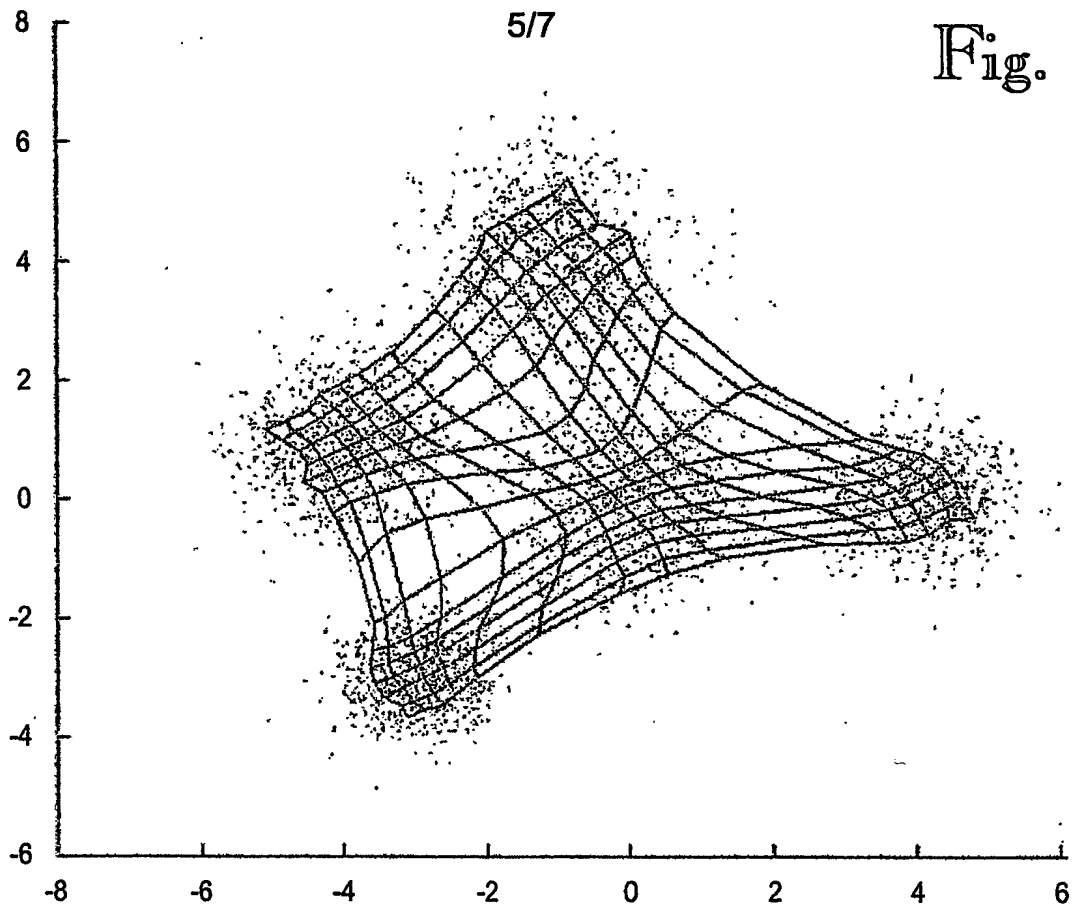


Fig. 9

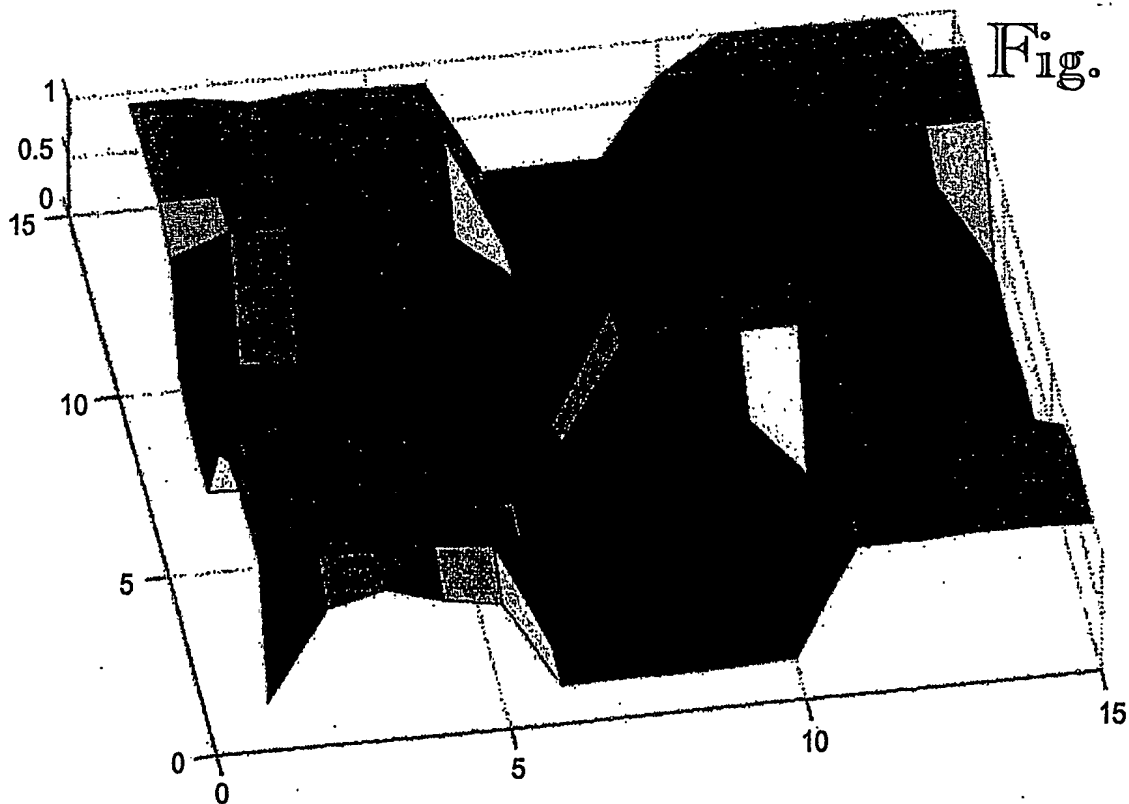


Fig. 10

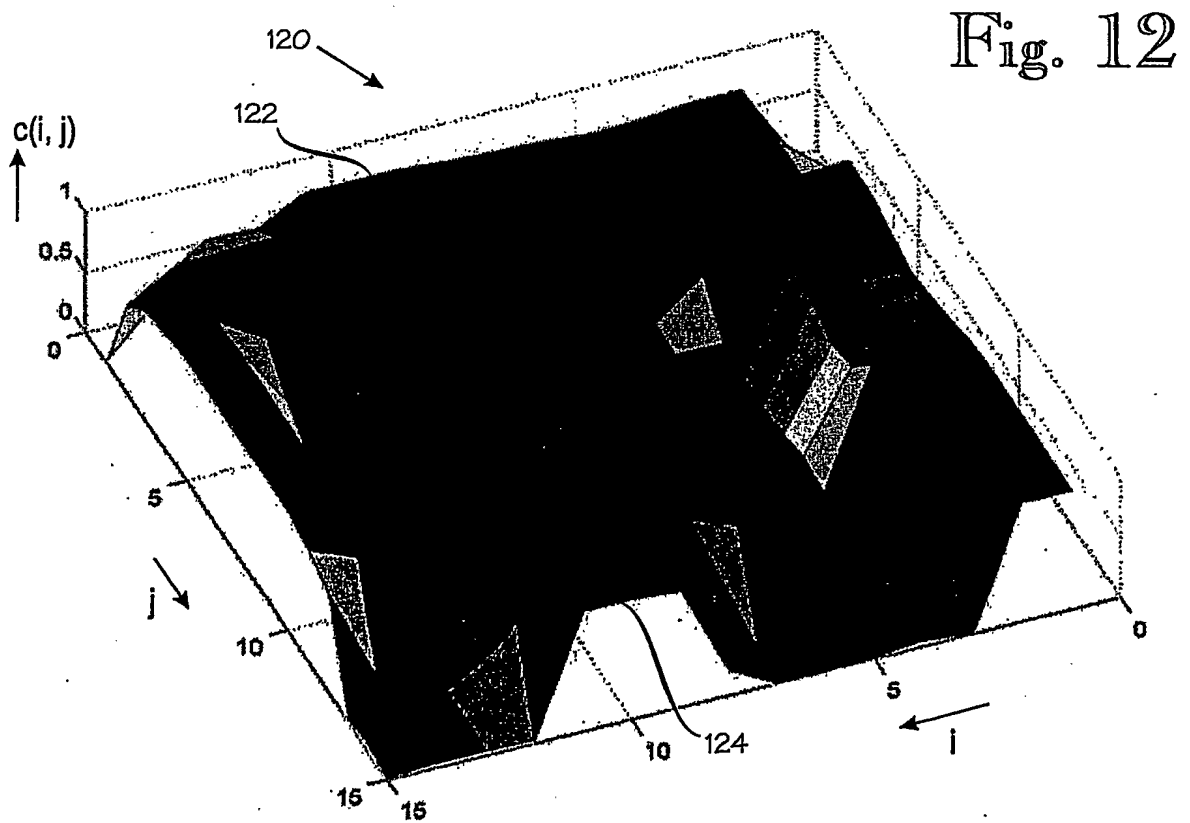
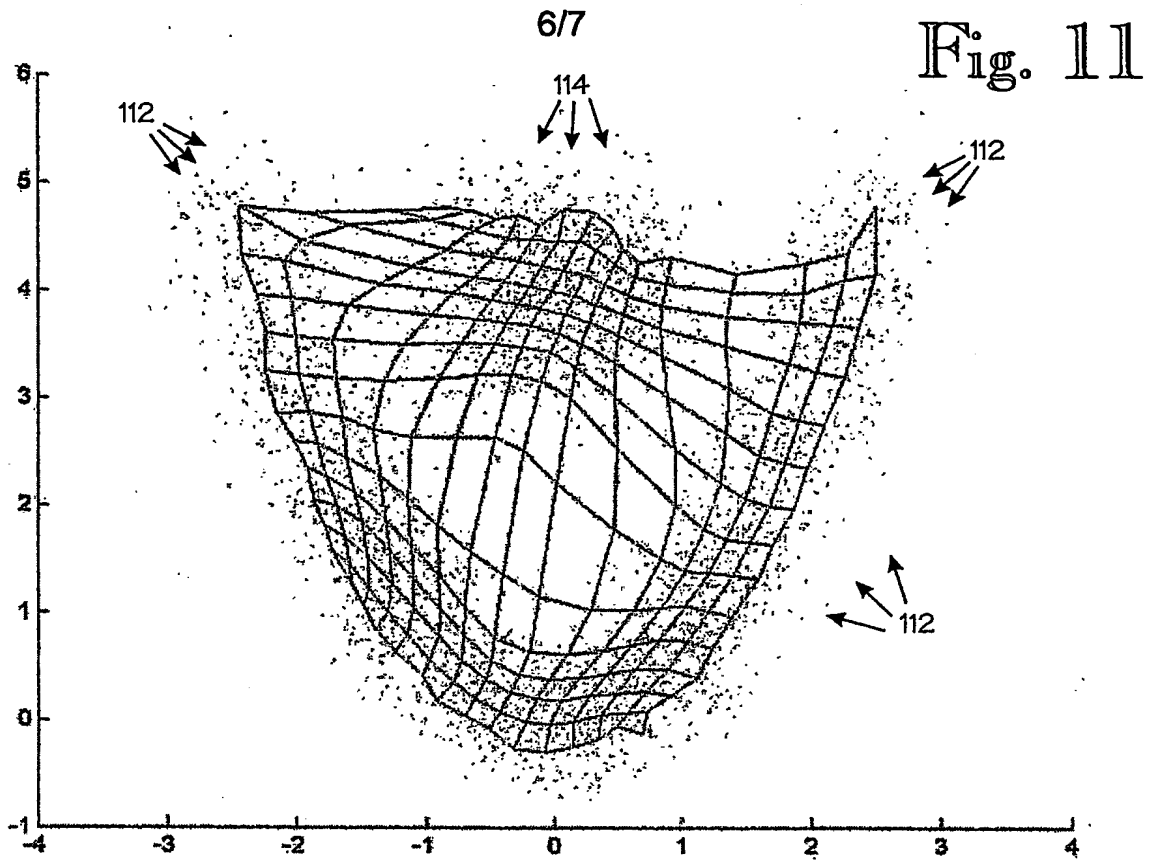


Fig. 13

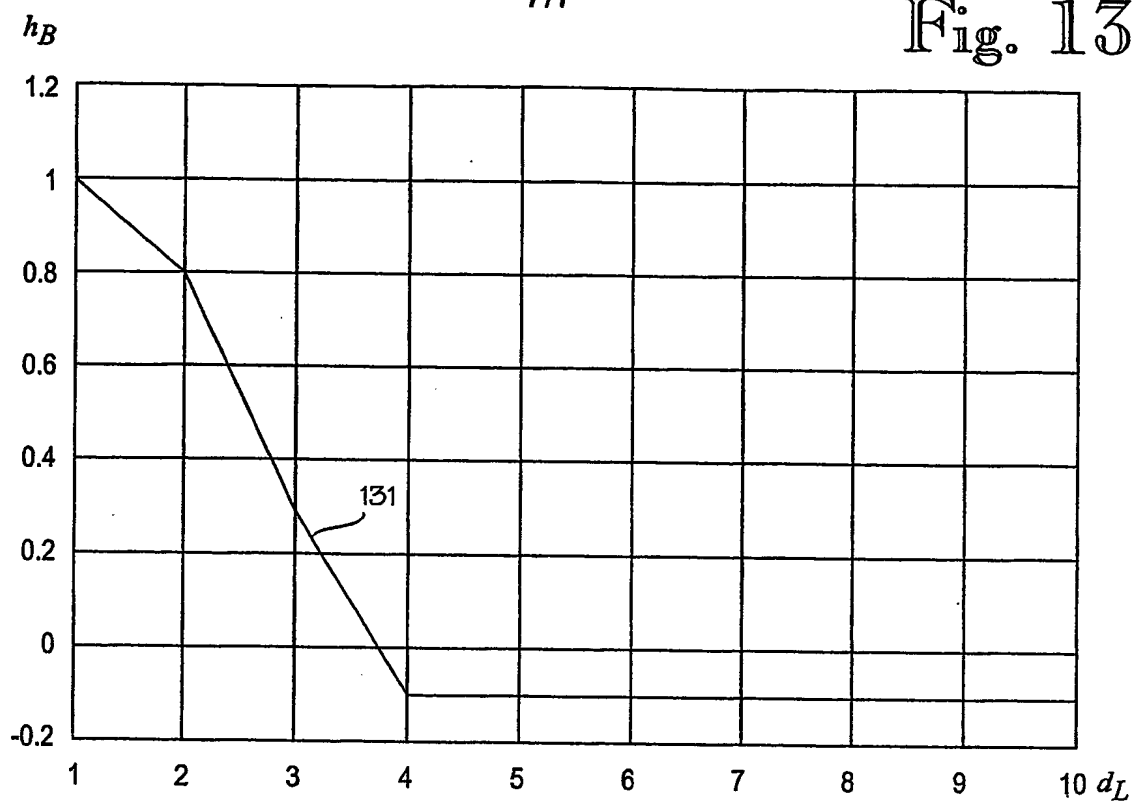
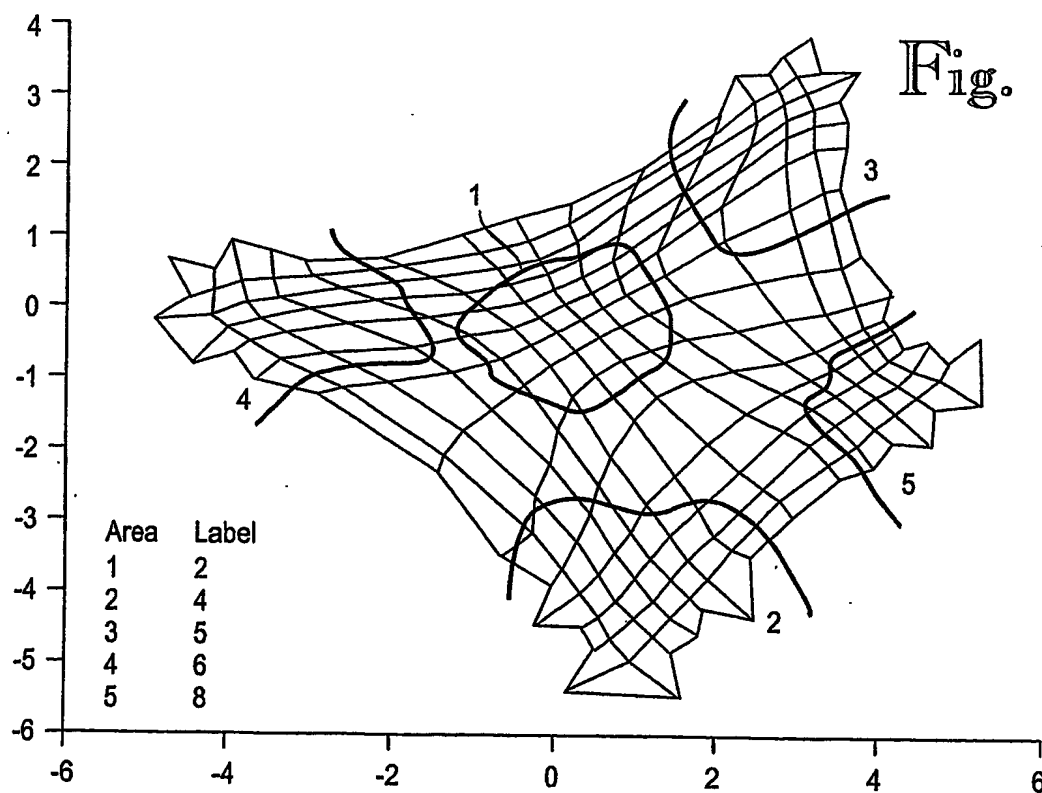


Fig. 14



**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.